**<u>AMENDMENTS TO THE CLAIMS:</u>**

The following listing of claims replaces all prior listings.

**<u>IN THE CLAIMS</u>**:

1. (Currently Amended) A method comprising:

allocating each received packet<u>, based on priority information in the received packet, </u>to at least one arrival queue <u>of a plurality of arrival queues, wherein each of the plurality of arrival queues handles packets based on a traffic class associated with a priority</u>, wherein each received packet comprises an internet protocol packet;

placing each packet in the allocated arrival queue if said arrival queue is not full, otherwise dropping said packet;

scheduling, by a scheduler coupled to the at least one arrival queue, packets from the arrival queue to at least one transfer queue;

responsive to transfer of a packet to a transfer queue, generating an interrupt;

responsive to receipt of an interrupt, allocating the packet from said transfer queue to one of a plurality of processor queues;

placing the packet in the allocated processor queue if said processor queue is not full, <u>wherein the at least one transfer queue and the at least one processor queue do not drop packets, and wherein the scheduler inhibits placement of the packet when the processor queue is full to prevent dropping the packet</u>~~ otherwise dropping said packet~~; and

scheduling packets from the processor queues to be processed, wherein the at least one arrival queue, the at least one transfer queue, and the plurality of processor

queues are separate queues, wherein the scheduler includes a first quantity N of inputs each corresponding to the at least one arrival queue, the scheduler further including a second quantity M of outputs each corresponding to the at least one transfer queue, wherein the second quantity M is less than or equal to the first quantity N.

2. (Currently Amended) A method comprising:

receiving a plurality of packets at one or more of a plurality of network devices coupled via one or more buses to at least one of a processor and a memory, wherein the packet comprises an internet protocol packet;

allocating each received packet to at least one arrival queue, wherein the one or more of the plurality of network devices comprises a scheduler, the at least one arrival queue, and at least one transfer queue, wherein the at least one of the processor and the memory further comprises at least one of a plurality of processor queues, wherein the scheduler is further coupled to the at least one transfer queue;

placing each packet in the allocated arrival queue if the at least one arrival queue is not full, otherwise dropping the packet;

scheduling, by the scheduler coupled to the at least one arrival queue, each packet from the arrival queue to the at least one transfer queue;

responsive to transfer of a packet to the at least one transfer queue, generating an interrupt;

responsive to receipt of the interrupt, allocating the packet from the at least one transfer queue to at least one of the plurality of processor queues;

placing the packet in the allocated processor queue if said queue is not full, wherein the at least one transfer queue and the at least one processor queue do not drop packets, and wherein the scheduler inhibits placement of the packet when the processor queue is full to prevent dropping the packet ~~otherwise dropping said packet~~; and

scheduling packets from the at least one of the plurality of processor queues to be processed, wherein the at least one arrival queue, the at least one transfer queue, and the plurality of processor queues are separate queues

3. (Previously Presented) A method according to claim 1, wherein at least one device has a plurality of arrival queues.

4. (Previously Presented) A method according to claim 3, wherein each arrival queue is associated with a traffic class, each packet being allocated to the at least one queue in accordance with the traffic class of each packet.

5. (Previously Presented) A method according to claim 4, wherein the traffic class is priority information embedded in the each packet.

6. (Previously Presented) A method according to claim 1, wherein at least one device comprises a plurality of transfer queues.

7. (Previously Presented) A method according to claim 1, wherein the number of transfer queues is less than the number of arrival queues.

8. (Previously Presented) A method according to claim 1, wherein the scheduling of packets from the arrival queue to the transfer queue is dependent upon one or more of: the traffic profile; the quality of service requirement; or the characteristics of the transfer queues.

9. (Previously Presented) A method according to claim 1, wherein the transfer queue comprises a device level transfer queue and a processor level transfer queue, wherein the device level transfer queue receives packets from the arrival queue, and the processor level transfer queue receives packets from the device level transfer queue.

10. (Previously Presented) A method according to claim 9, wherein packets are transferred to the processor level transfer queue from the device level transfer queue whenever there is space in the processor level transfer queue.

11. (Previously Presented) A method according to claim 10, wherein packets are never dropped from the transfer queue.

12. (Previously Presented) A method according to claim 1, wherein the processor queues are associated with different priorities.

13. (Previously Presented) A method according to claim 12, wherein the highest priority queue has the lowest drop probability and the lowest latency.

14. (Previously Presented) A method according to claim 1, wherein responsive to receipt of the interrupt, a packet is removed from a transfer queue and classified.

15. (Previously Presented) A method according to claim 14, wherein the classification is based on a determination of priority.

16. (Previously Presented) A method according to claim 14, wherein the packet is allocated to a processor queue in accordance with a classification of the packet.

17. (Previously Presented) A method according to claim 14, wherein the packet is placed in the allocated processor queue if said queue is not full, otherwise the packet is dropped.

18-34 (Cancelled)

35. (Currently Amended) An apparatus comprising:

a processor configured to allocate a received packet to at least one arrival queue, wherein the received packet comprises an internet protocol packet, wherein the processor is configured to place each packet in the allocated queue if said arrival queue is not full, otherwise dropping said packet, wherein the processor is configured to schedule packets from the arrival queue to at least one transfer queue, wherein the processor is responsive to transfer of a packet to a transfer queue, configured to generate an interrupt, wherein the processor is responsive to receipt of an interrupt, configured to allocate the packet from said transfer queue to one of a plurality of processor queues, wherein the processor is configured to place the packet in the allocated processor queue if said processor queue is not full, wherein the at least one transfer queue and the at least one processor queue do not drop packets, and wherein the scheduler inhibits placement of the packet when the processor queue is full to prevent dropping the packet, and wherein the processor is configured to schedule packets from the processor queues to be processed, wherein the at least one arrival queue, the at least one transfer queue, and the plurality of processor queues are separate queues, wherein the scheduler includes a first quantity N of inputs each corresponding to the at least one arrival queue, the scheduler further including a second

quantity M of outputs each corresponding to the at least one transfer queue, wherein the second quantity M is less than or equal to the first quantity N.

36. (Previously Presented) The apparatus according to claim 35 further comprising a plurality of arrival queues.

37. (Previously Presented) The apparatus according to claim 36, wherein each arrival queue is associated with a traffic class, each packet being allocated to at least one queue by the processor in accordance with the traffic class of each packet.

38. (Previously Presented) The apparatus according to claim 35 further comprising a plurality of transfer queues.

39. (Previously Presented) The apparatus according to claim 35, wherein the transfer queue comprises a device level transfer queue and a processor level transfer queue, the device level transfer queue configured to receive packets from the arrival queue, and the processor level transfer queue configured to receive packets from the device level transfer queue.

40. (Previously Presented) The apparatus according to claim 39, wherein packets are transferred to the processor level transfer queue from the device level transfer queue whenever there is space in the processor level transfer queue.

41. (Previously Presented) The apparatus according to claim 40, wherein packets are never dropped from the transfer queue.

42. (Previously Presented) The apparatus according to claim 35, wherein the processor queues are configured to be associated with different priorities.

43. (Previously Presented) The apparatus according to claim 35, wherein the processor is configured responsive to receipt of the interrupt, to remove a packet from a transfer queue, and to classify the packet.

44. (Previously Presented) The apparatus according to claim 35, wherein the processor is configured to allocate the packet to a processor queue in accordance with a classification of the packet.

45. (Previously Presented) The apparatus according to claim 44, wherein the packet is placed in the allocated processor queue if said queue is not full, and otherwise the packet is dropped.

46. (Currently Amended) A non-transitory computer-readable storage medium encoded with instructions that, when executed on a computer, perform a process, the process comprising:

allocating each received packet to at least one arrival queue, wherein each received packet comprises an internet protocol packet;

placing each packet in the allocated arrival queue if said arrival queue is not full, otherwise dropping said packet;

scheduling, by a scheduler coupled to the at least one arrival queue, packets from the arrival queue to at least one transfer queue;

responsive to transfer of a packet to a transfer queue, generating an interrupt; responsive to receipt of an interrupt, allocating the packet from said transfer queue to one of a plurality of processor queues;

placing the packet in the allocated processor queue if said processor queue is not full, <u>wherein the at least one transfer queue and the at least one processor queue</u>

do not drop packets, and wherein the scheduler inhibits placement of the packet when the processor queue is full to prevent dropping the packet; and

scheduling packets from the processor queues for processing, wherein the at least one arrival queue, the at least one transfer queue, and the plurality of processor queues are separate queues, wherein the scheduler includes a first quantity N of inputs each corresponding to the at least one arrival queue, the scheduler further including a second quantity M of outputs each corresponding to the at least one transfer queue, wherein the second quantity M is less than or equal to the first quantity N.